Vector Convolution

- Let $u = (u_0, \ldots, u_{m-1})$ and $v = (v_0, \ldots, v_{n-1})$ be vectors of length m and n, respectively
- The convolution of u and v, denoted $u * v$, is a vector of length $m + n - 1$ with component k, $0 \leq k < m + n - 1$, equal to
$$\sum_{(s,t)\in\{0,\ldots,m-1\}\times\{0,\ldots,n-1\}\,:\,s+t=k} u_s v_t$$
- Numerous practical applications
- We'll focus on the connection to polynomial multiplication

Connection to Polynomial Multiplication

- Let $A(x) = \sum_{0 \leq k < n} a_k x^k$ and $B(x) = \sum_{0 \leq k < n} b_k x^k$ be two polynomials of degree less than n
- The product of $A(x)$ and $B(x)$ is a polynomial $C(x)$ of the form $\sum_{0 \leq k < 2n-1} c_k x^k$ where

$$c_k = \sum_{s,t \in \{0,\ldots,n-1\} \, : \, s+t=k} u_s v_t$$

  is component k of $a * b$
- Thus the task of computing the coefficients of $C(x)$ is equivalent to the task of computing $a * b$

Polynomial Evaluation and Interpolation

- Given $n+1$ data points $(x_0, y_0), \ldots, (x_n, y_n)$ where the x_k's are all distinct, there is a unique polynomial $p(x)$ of degree at most n such that $p(x_k) = y_k$ for $0 \leq k \leq n$
- We can think of such a set of data points as providing a "point-based" representation of the polynomial
- A framework for polynomial multiplication
  - Pick a set S of at least $2n-1$ distinct x_k values
  - Evaluate $A(x)$ (resp., $B(x)$) on all values in S to obtain a point-based representation
  - For each x in S, determine $C(x)$ by multiplying $A(x)$ and $B(x)$; this gives a point-based representation of $C(x)$
  - Recover the coefficients of $C(x)$ by polynomial interpolation

Some Useful Facts about Complex Numbers

- $\exp(i\theta) = \cos\theta + i\sin\theta$, i.e., the complex number with real part $\cos\theta$ and imaginary part $\sin\theta$
  - Lies on the unit circle in the complex plane, since $\cos^2\theta + \sin^2\theta = 1$
  - Lies at angle θ (counterclockwise) from the positive real axis
- For any positive integer N, let ω_N denote $\exp(2\pi i/N)$
  - Thus, for any integer k, ω_N^k lies on the unit circle in the complex plane, at angle $2\pi k/N$ from the positive real axis
  - Hence $(\omega_N^k)^N = 1$

Some Useful Facts about Complex Numbers (cont'd)

- For any integer k, ω_N^k is a root of the polynomial $x^N - 1$
- Observe that $x^N - 1 = (x-1)\sum_{0 \leq s < N} x^s$
- For any integer k that is not a multiple of N, we have $\omega_N^k \neq 1$
- Hence for any integer k that is not a multiple of N, ω_N^k is a root of the polynomial $\sum_{0 \leq s < N} x^s$ (Claim 1)
- For any positive integer N, let S_N denote the set of N complex numbers $\{\omega_N^k \mid 0 \leq k < N\}$

Fast Polynomial Evaluation

- Let $A(x) = \sum_{0 \leq k < n} a_k x^k$ and assume that n is a power of 2
- Let $A_{even}(x)$ denote $a_0 + a_2 x + a_4 x^2 + \ldots + a_{n-2} x^{(n/2)-1}$ and let $A_{odd}(x)$ denote $a_1 + a_3 x + a_5 x^2 + \ldots + a_{n-1} x^{(n/2)-1}$
- Observe that $A(x) = A_{even}(x^2) + x A_{odd}(x^2)$
- Observe that $\{x^2 \mid x \in S_{2n}\} = S_n$
- These observations imply a recursive algorithm for evaluating $A(x)$ at every element of S_{2n}
- Mergesort-like recurrence yields $O(n \log n)$ time bound

Polynomial Interpolation

- We wish to compute the coefficients of a polynomial $C(x) = \sum_{0 \leq s < 2n} c_s x^s$ given the value of $C(x)$ at each element of S_{2n}
  - Remark: In our application to fast polynomial multiplication, we happen to know that $c_{2n-1} = 0$ since $C(x)$ has degree at most $2n - 2$
- Let $D(x)$ denote the polynomial $\sum_{0 \leq s < 2n} C(\omega_{2n}^s) x^s$
- Key Claim: For any integer k such that $1 \leq k \leq 2n$, we have $D(\omega_{2n}^k) = 2nc_{2n-k}$

Proof of the Key Claim

- For any k such that $1 \leq k \leq 2n$, we have

$$\begin{aligned}
D(\omega_{2n}^{k}) &= \sum_{0 \leq s < 2n} C(\omega_{2n}^{s}) \omega_{2n}^{ks} \\
&= \sum_{0 \leq s < 2n} \left(\sum_{0 \leq t < 2n} c_t \omega_{2n}^{st} \right) \omega_{2n}^{ks} \\
&= \sum_{0 \leq s < 2n} \sum_{0 \leq t < 2n} c_t \omega_{2n}^{(k+t)s} \\
&= \sum_{0 \leq t < 2n} c_t \sum_{0 \leq s < 2n} \left(\omega_{2n}^{k+t} \right)^{s}
\end{aligned}$$

Proof of the Key Claim (cont'd)

- For any k such that $1 \leq k \leq 2n$, we have

$$D(\omega_{2n}^{k}) \;=\; \sum_{0 \leq t < 2n} c_t \sum_{0 \leq s < 2n} \left(\omega_{2n}^{k+t}\right)^{s}$$

- When $t = 2n - k$, each term in the inner sum is 1, so the inner sum is $2n$
- What if t belongs to $\{0, \ldots, 2n-1\} \setminus \{2n-k\}$?
  - $k + t$ is not a multiple of $2n$
  - By Claim 1, ω_{2n}^{k+t} is a root of the polynomial $\sum_{0 \leq s < 2n} x^s$
  - Hence the inner sum is zero

Fast Polynomial Interpolation

- As we have seen earlier, we can evaluate $D(x)$ for every element of S_{2n} in $O(n \log n)$ time
- Since $\omega_{2n}^{2n} = \omega_{2n}^{0} = 1$, this gives us $D(\omega_{2n}^{k})$ for all k such that $1 \leq k \leq 2n$
- The key claim implies that we can use these $2n$ values to obtain all of the coefficients of $C(x)$ in $O(n)$ time

The Discrete Fourier Transform

- The discrete Fourier transform (DFT) maps any given vector $a = (a_0, \ldots, a_{N-1})$ of complex numbers to the vector $(A(\omega_N^0), \ldots, A(\omega_N^{N-1}))$ where $A(x) = \sum_{0 \leq k < N} a_k x^k$
- For N a power of 2, we can use the foregoing recursive approach to compute the DFT of such a vector in $O(N \log N)$ time
- This algorithm for computing the DFT is called the Fast Fourier Transform (FFT)
  - While we have focused on the special case where N is a power of 2, it is possible to generalize the FFT to handle arbitrary N efficiently